cadence®

Incisive Enterprise Specman Products

Verification automation from block to chip to system levels

Part of the Cadence® Incisive® functional verification platform, Incisive Enterprise Specman® products blend leading-edge process automation technology with the comprehensive Plan-to-Closure Methodology to simplify and speed verification. Specman products automate the entire verification process, from individual blocks to full chips to the project level. With Specman technology, designers benefit from increased productivity and a predictable path to high-quality silicon.

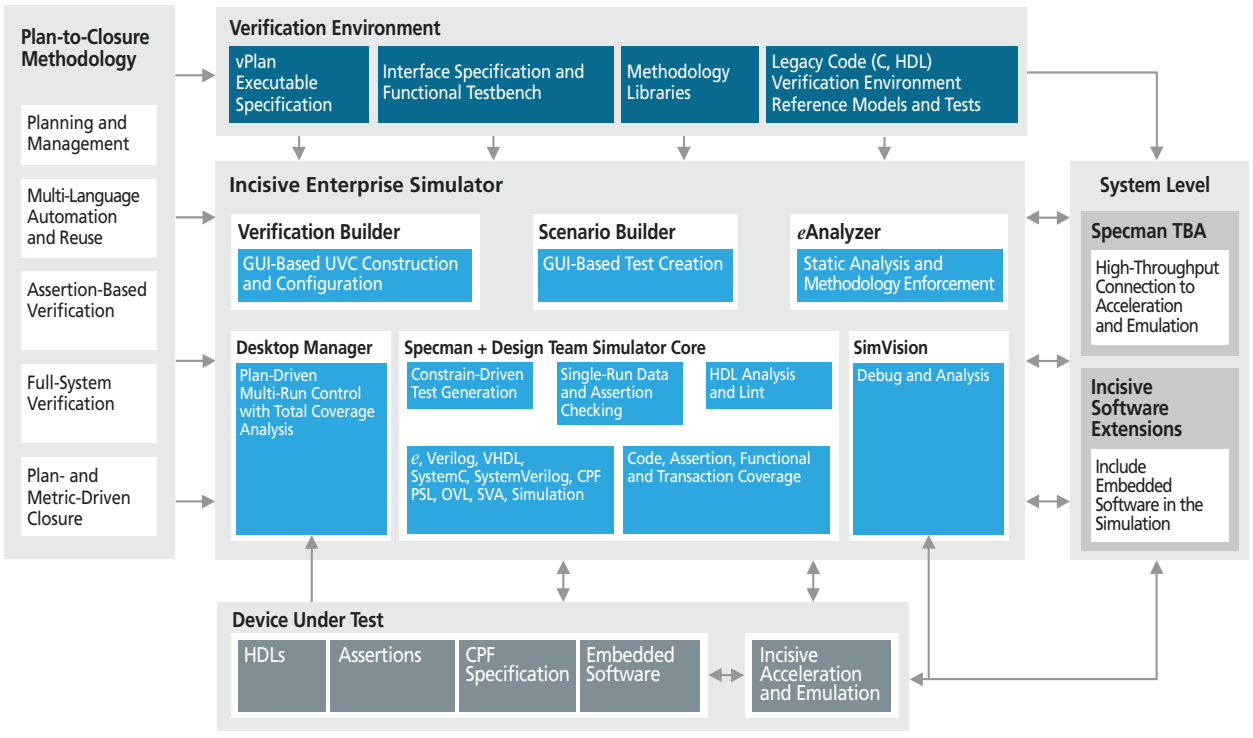

Figure 1: Incisive Enterprise Specman Elite® Testbench helps drive the entire process of verification at block, chip, system, and project levels. The Verification Builder, Scenario Builder, and eAnalyzer components automate the creation and configuration of reusable, scalable, eRM-compliant environments and stimuli. Incisive Software Extensions offer a high-throughput channel between the testbench and the device under test (DUT), and enable access to embedded software as if it were another part of the DUT.

Incisive Enterprise Specman Elite Testbench

Successful verification of today's multi-million-gate designs requires optimal speed and efficiency. But verification teams often struggle to squeeze in enough cycles to ensure that functional bugs won't surface in silicon. Incisive Enterprise Specman Elite Testbench is a comprehensive environment that accelerates and simplifies all aspects of verification (automatic generation of functional tests, data and assertion checking, and functional coverage analysis) in addition to supporting the production-proven Plan-to-Closure Methodology.

Verification teams can extend the functionality of Specman technology with Incisive Enterprise Specman ESL Testbench (Specman ESL), which provides a high-throughput channel between the testbench and the device under test (DUT), and enables Plan-to-Closure verification automation of embedded software exactly as if it were another part of the DUT. With other elements from the Incisive platform, including verification IP, hardware acceleration and emulation, analog/mixed-signal/RF verification, and formal assertion verification, Specman products support any testbench, HDL, software, or assertion IP.

Many engineers already create testbenches in C, VHDL, Verilog®, and SystemVeriog, and may have invested time and effort in internal solutions. But, realistically, these tools are rewritten project to project without allowing for significant reuse. Nor do they contain the engines and aspect-oriented programming (AOP) support already built into Specman products, such as the patented constraint solver that generates stimulus automatically.

With Specman products, engineers can use the powerful *e* verification language to capture rules from executable specifications and use the information to automate verification. The Specman methodology finds even the most subtle, corner-case bugs because it eliminates misrepresentations of specifications.

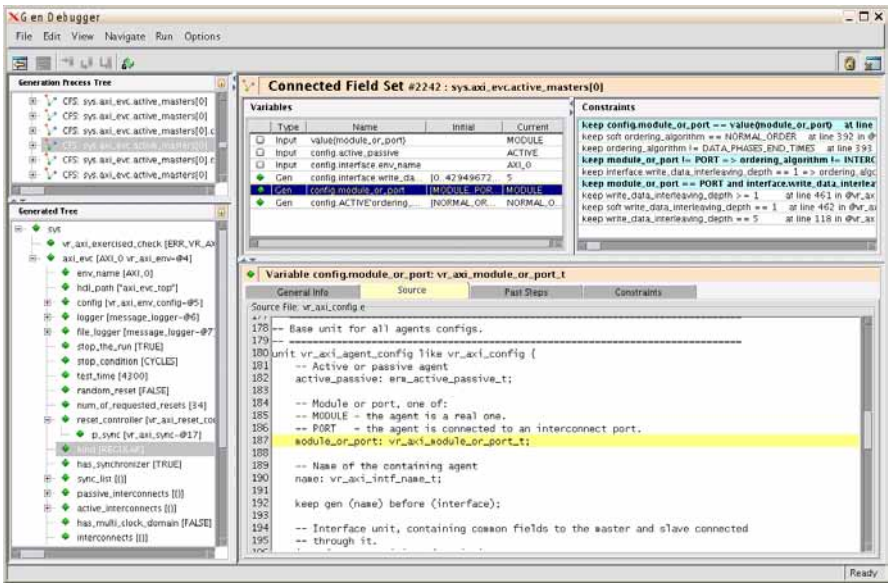

Figure 2: Specman "IntelliGen" technology simultaneously increases performance, scalability, and control over stimuli. The Generation Debugger confirms that constraints are set to drive Specman toward suspected corner cases.

Benefits

- Captures executable specifications and eliminates misrepresentations
- Leverages the *e* language's unique AOP capabilities
- "IntelliGen" constraint solvers automate test generation with up to 5x faster runtime, unprecenteded distribution control, and scalability
- Speeds debugging via automatic data and assertion checking
- Increases predictability with functional coverage analysis
- Supports all IEEE-standard languages including *e*, SystemC®, C, C++, VHDL, Verilog, and SystemVerilog
- Supports built-in IP reuse to leverage existing investments in verification IP
- Works with all major simulators

Features

Constraint-driven stimulus generation

Specman products provide constraint-driven test generation that automates the process of generating functional verification tests. By specifying constraints, engineers can target the generator quickly and easily to create any test in their functional test plan. They can even generate tests on-the-fly based on the current design state, making it possible to detect hard-to-reach corner cases.

Data and assertion checking

Powerful temporal constructs allow verification specialists and designers to capture complex protocols for assertion checking. On-the-fly data checking and generation provides context-specific expected values. With Specman products, verification engineers can use any combination of gray-, black-, or white-box checking to speed debugging.

Functional coverage analysis

An executable functional test plan measures the progress of verification, and functional analysis automatically identifies holes in the test coverage. Since functional coverage is a meaningful and direct measure of the completeness of verification, functional coverage analysis increases predictability in verification schedules.

Rapid creation of libraries of reusable tests

Specman products fully support the testbench reuse component of the Plan-to-Closure Methodology, which describes how to create reusable verifi-

cation components in any IEEE-standard language, and provides guidelines for setting up multi-language interfaces to existing IP for maximum operational flexibility. The process is based on the time-tested *e* Reuse Methodology (*e*RM) and System Verification Methodology (SVM).

With the Verification Builder component, users can quickly create and configure reusable verification components in *e*RM-compliant format. Verification Builder also supports plug-ins that help users automatically connect the DUT to the testbench. A similar plug-in for Incisive Software Extensions supports rapid binding of the testbench to the processor model(s).

The Scenario Builder component helps designers create test cases in a fraction of the time it would take to write them in a hardware verification language, eliminating the need to learn a verification language and object-oriented programming. Users simply drag-and-drop randomized verification elements and adjust their constraints to build a scenario. Scenario Builder makes it just as easy to build complex system-level test cases: users define stimuli for each interface that can be synchronized on existing or user-defined events.

When combined with the "sequence" feature of the Plan-to-Closure Methodology, stimuli that exercise corner-case scenarios at the block level can be reused at the system level to verify how the entire chip behaves in that corner case. Furthermore, users can graphically create these sequences as easily as test cases, which they can use in other tests hierarchically or in other sequences by dragging and dropping in a new scenario. All scenarios are written out into a reusable format, supporting the creation of portable libraries of protocol checkers and other structured behaviors for reuse on future projects, or system-level verification of the same project.

Testbench static analysis

Static analysis catches testbench bugs and coding surprises early in the verification cycle. It performs more than 200 checks to flag syntactic, semantic, and functional errors. A flow that includes testbench analysis before simulation will check the code for reusability per *e*RM-compliance rules, testbench performance issues, race conditions, pre-defined coding style rules, generation constraints, and semantic ambiguities. These rules can be expanded to include corporate style guidelines. The result: with the powerful rule-definition GUIs and graphical analysis tools, engineers write working code correctly the first time.

HDL simulator interfaces

Specman products integrate with all leading HDL simulators and support a high-performance, direct kernel interface to all Incisive simulators. Users can sample and drive internal signals of the DUT. With 100% controllability and observability of otherwise inaccessible internal signals, all Specman engines have full access to signal values during simulation.

Transaction-level modeling and SystemC support

Specman products provide SystemC interface mechanisms to drive and monitor transaction-level models (TLMs) as well as signal-level models. You can apply Specman verification methodologies to the verification of SystemC architectural models using TLMs and reference models including mixed SystemC/RTL environments, and co-verify SystemC models used for software development.

In addition to supporting Incisive simulators, Specman products provide interface adaptors for SystemC simulators including OSCI® and CoWare ConvergenSC. With Specman technology, engineers can create a single verification environment to verify their SystemC model and then reuse it throughout the entire downstream flow, from RTL simulation to acceleration and emulation.

HW/SW co-verification

Specman products support all leading hardware/software co-verification tools. They also integrate seamlessly with Incisive Software Extensions in the Specman ESL co-verification environment to enable functional testing of both hardware and software. Early integration and debugging of HW/SW systems eliminates errors and shortens time to market for the combined system.

Plan-To-Closure Methodology

The Plan-to-Closure Methodology provides a system of best-known principles, guidelines, and procedures that increase project productivity and predictability, and ensure overall system-level quality. It includes documented best practices, golden examples that serve as templates to help engineers learn and apply the methodology, and libraries (code building blocks and utilities) that automate the process and eliminate many redundant verification-coding tasks.

In addition, unlike any other commercially available methodology, the Plan-to-Closure "Knowledge System" (a web-based portal technology) allows you to easily research topics of interest, and then customize the methodology to your specific needs. The Plan-to-Closure Methodology spans verification planning and management, assertion-based verification, testbench automation and reuse, and system-level verification.

Incisive Enterprise Specman ESL Testbench

Traditionally, the ESL acronym has been used to describe the need for scalability and increased abstraction to accommodate growing HDL designs. Incisive Enterprise Specman ESL Testbench (Specman ESL) builds on this traditional flow to enable systems engineers, logic designers, software engineers, verification engineers, and system validation teams to do "in-system" verification at the block, chip, and full system levels.

This is all made possible by using the components in Specman ESL to leverage the metric-driven Plan-to-Closure Methodology to span design and verification—from an initial system specification and system verification plan to full system-level integration and closure. The result: rapid discovery of deeply buried, cross-domain bugs that are impossible to find otherwise.

Incisive software extensions

Incisive Software Extensions give your testbench access to embedded software exactly as if it were another part of the HDL DUT. Using extensions to the familiar Incisive SimVision debug tool, engineers can simultaneously control and verify software methods, procedures, variables, registers, and other elements with a traditional hardware-centric DUT using the same time-tested coverage-driven verification (CDV) process described in the Plan-to-Closure Methodology. Furthermore, Incisive Software Extensions rises above hardware/software co-verification limitations by supporting processor models in any form: workstation-based host-code execution, Instruction Set Simulator (ISS), full RTL CPU models, hardware acceleration and emulation, and even prototype silicon.

Transaction-based accelerated (TBA)

Transaction-based acceleration (TBA) technology utilizes message-level communication between the Specman testbench and the acceleration/emulation hardware. By communicating at the message level instead of the signal level, "TBA Specman" significantly reduces the amount of testbench-DUT communication, greatly increasing overall throughput over simulation alone (up to 100x when used in combination with the Incisive Palladium® or Xtreme® series of hardware accelerators and emulators). This level of performance enables full system-level validation that includes embedded and application-layer software.

When TBA Specman and Incisive Software Extensions are combined with the verification plan ("vPlan")-driven Plan-to-Closure flow, the result is a predictable, metric-driven process for both hardware and software development based on an executable plan that enforces adherence to the system specification. In short, this real-time, automatic annotation of metrics against the specification gives engineers the information they need to confidently argue for additional time to explore more high-risk corner cases, or to ask for a new pen to autograph the signoff form.

Specifications

Language support

- Testbench
  - *e* (IEEE 1647)
  - Interface to SystemVerilog (IEEE 1800) testbenches (this is a high-performance, direct kernel interface when using Incisive platform simulators)
- Device under test
  - Verilog (IEEE 1364-1995 and IEEE 1364-2001 extensions)
  - SystemVerilog (IEEE 1800)
  - VHDL (IEEE 1076-1987, IEEE 1076-1993, IEEE 1076.4-2000 (VITAL 2000))
  - SystemC (OSCI SystemC v2.2, IEEE 1666)
  - PSL (IEEE 1850)
  - SVA (IEEE 1800)
  - C and C++ models
  - Matlab models
  - Analog models in Verilog-A, VHDL-A, or SPICE formats
  - Post-silicon hardware
  - Specman ESL supports embedded software and high-throughput connections to accelerated and emulated DUTs

Testbench analysis

- 200+ checks to lint and analyze code for:
  - Code reusability as per *e* Reuse Methodology (*e*RM) compliance rules
  - Performance analysis
  - Race conditions
  - Pre-defined coding style rules
  - Generated constraints
- Graphical interface to sort, filter, and analyze messages with source code

Verification builder

- GUI support for configuration of Plan-to-Closure Universal Verification Components (UVCs)
- Outputs e (IEEE 1647) or SystemVerilog (IEEE 1800) Plan-to-Closure UVCs

Scenario builder

- Outputs stimulus conforming to the "sequences" capability defined in the IEEE 1647-based *e*RM

Support For Cadence IP

- Design IP
  - Cadence SoC Functional Verification Kit
- Verification IP
  - Supports all simulation-based Universal Verification Components (UVCs), transaction-based VIP, assertion-based VIP, and SpeedBridge® rate adapters used in emulation
  - Supports the full portfolio of Cadence UVCs: PCI Express, AHB, AXI, AMBA™, USB, and Ethernet, PCI, SATA, and OCP
  - UVCs are designed to use all elements of the Incisive platform and feature a comprehensive Compliance Management System that leverages "vPlans" to exhaustively verify protocol compliance

Third-party support

- Models
  - Third-party model support through the Cadence Verification IP Partner program
- Software
  - Third-party software support through the Connections® program with more than 30 verification company partners

Interfaces

- Direct kernel interface to all Incisive platform simulators
- Direct C language interface
- Socket interface
- PLI (IEEE 1364)
- DPI (IEEE 1800)
- VPI (PLI 2.0, IEEE 1364)
- VHPI

Platforms

- Sun Solaris
- HP-UX
- Linux

Cadence Services and Support

- Cadence application engineers can answer your technical questions by telephone, email, or Internet—they can also provide technical assistance and custom training
- Cadence certified instructors teach more than 70 courses and bring their real-world experience into the classroom
- More than 25 Internet Learning Series (iLS) online courses allow you the flexibility of training at your own computer via the Internet
- Cadence Online Support gives you 24x7 online access to a knowledgebase of the latest solutions, technical documentation, software downloads, and more

cadence®

Cadence is transforming the global electronics industry through a vision called EDA360. With an application-driven approach to design, our software, hardware, IP, and services help customers realize silicon, SoCs, and complete systems efficiently and profitably. www.cadence.com

21509 11/11 MK/DM/PDF